(2)

AD-A240 478

Software Technology for Adaptable, Reliable Systems (STARS)

Submitted to:
Electronic Systems Division:
Air Force Systems Command, USAF:
Hanscom AFB, MA 01731-5000:

Contract No:
F-19628-88-D-0028

CDRL 00320 Standards and Guidelines For Repository

March 17, 1989

The Boeing Company
Boeing Aerospace Division
Engineering Technology
P.O. Box 3999
Seattle, Washington 98124

91-10153

Approved for public release - distribution is unlimited

051

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	17-MAR-89	

4. TITLE AND SUBTITLE
Standards and Guidelines for Repository

5. FUNDING NUMBERS
C: F19628-88-D-0028

6. AUTHOR(S)
Margaret J. Davis

TA: BQ-12

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The Boeing Company
Boeing Aerospace and Electronics Division
Systems and Software Engineering
P.O. Box 3999
Seattle, Washington 98124

8. PERFORMING ORGANIZATION REPORT NUMBER
D-613- 10320

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
ESD/AVS
Bldg. 17-04
Room 113
Hanscom Air Force Base, 01731-5000

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release - distribution unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT *(Maximum 200 words)*

This technical report contains recommendations for guidelines and standards to be used in developing Ada programs and technical documents for delivery to the repository.

14. SUBJECT TERMS
Keywords: STARS
style guide
repository

15. NUMBER OF PAGES
39

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	None

Standards and Guidelines for Repository

Prepared by

Margaret J. Davis 3/16/89
Margaret J. Davis, Chief
Programmer, Q12 – Seattle

Reviewed by

James E. King
System Architect

Reviewed by

John M. Neorr 3/16/89
Development Manager

Approved by

W. M. Hodges 3/16/89
William M. Hodges
STARS Program Manager

ABSTRACT

This technical report contains recommendations for guidelines and standards to be used in developing Ada programs and technical documents for delivery to the repository.

KEY WORDS

style guide
repository

DTIC COPY INSPECTED 1

A-1

PREFACE

This document conforms to Boeing standards and guidelines

Notice: The contents of this document is available in electronic form on the STARS Repository in Wichita, Kansas. It may be found in the Boeing directory for increment one deliveries under Q Task 12: Guidelines and Repository.

TABLE OF CONTENTS

1. SCOPE

1.1 Identification

This technical report makes recommendations for guidelines and standards to be adopted for items delivered to the STARS repository. The guidelines and standards cover Ada source code, including specifications, and technical documentation.

1.2 Purpose

The STARS repository is "to facilitate the distribution and sharing of software and documentation including interim and final results....Delivery of software will not be considered complete until code, along with confirmation of its compilation by the prime-contractor-designated peer review using a validated DoD compiler, has been made available for a designated STARS repository." Guidelines and standards for software and documentation, if used, simplify the sharing of textual and graphical material. To this end, the repository will not only provide guidelines and standards but will also acquire tools that assist the repository users' in preparation of items according to these standards and assist the repository administration in enforcement of these standards.

1.3 Introduction

Guidelines and standards not only impose a uniformity on the format of material but also provide standards that should enhance the reusability and portability of code and provide standards for content that can be used to identify delivered items for later retrieval.

Several existing standards are or may be applicable to setting up and enforcing guidelines for the repository. These are: (1) Standardized General Markup Language (SGML), which is a system for writing grammars that describe the structure of text material, (2) Ada Language Specification, (3) IGES, which is a standard for simple graphics, (4) Information Resource Dictionary System (IRDS), which is a standard for description of an information model that may be useful for interchange of designs that are hybrids of text and graphical material, (5) Postscript, which is a standard for page description, and (6) X Windows, which is an evolving standard for terminal and workstation display.

2. REFERENCED DOCUMENTS

This section lists pertinent material directly and in-directly referenced in this document.

[1] STARS Repository Preliminary System Description, 1987.

[2] CDRL 00670, "General S/W Development Plan", The Boeing Company. December 15, 1988.

[3] SPC-TR-87-105, "Ada Style Guide", Software Productivity Consortium.

3. Guidelines and Standards for Ada Source

The intention for source code delivered to the repository is that it be reusable on other projects and programs. At a minimum, to be reusable the source code must compile correctly. However, compilation does not guarantee that the code is correct, nor that someone else can understand what the code is used for or how to port the code to different compilation systems or different operating systems. Style guides provide recommendations that help Ada code be more portable and reusable. Standard prologues can be used to gather information that can be used to make the code and context in which the code is to be used more understandable.

3.1 Ada Source Code Style Guidelines

The recommendations for style guidelines to be followed for Ada source items delivered to the repository are identical to those found in the Boeing STARS General S/W Development Plan, CDRL 00670. These guidelines extend those given in the Software Productivity Consortium's (SPC) Ada Style Guidelines. The SPC Ada Style Guidelines are available on-line at the repository under the general information directory. The Boeing extensions include:

- Ada language keywords shall be lowercase.
- The first character of each word in an identifier name shall be capitalized; all other characters shall be in lowercase.
- A single space shall be placed before and after each delimiter.
- Three spaces shall be used to align control structures, continuation lines, and embedded units.
- A single blank line shall be used to separate logically related lines of text.
- The maximum line length shall be 80 characters.
- Associate identifiers with all loops.
- Associate identifiers with all nested blocks.
- All compilation units shall contain the standard prologue.
- Append the string "_Type" to all user defined types, subtypes, and derived types.

Tools that fully support or enforce the use of these guidelines do not yet exist. Applicable technology includes prettyprinters, context-sensitive, structure, and stream editors.

3.2 Ada Design Language (ADL) Guidelines

The recommendations for style and format guidelines for ADL are identical to those found in the Boeing STARS General S/W Development Plan, CDRL 00670. All ADL statements have the form '--| text', which allows the ADL to be treated by an Ada compiler as a comment.

There are standard ADL statements bundled together as a prologue (see following section for definition) to provide information about design goals, implementation details, revision history, classification, and other commentary needed to facilitate reuse.

3.3 Prologue Guidelines

The standard prologues have been designed to capture information needed for reuse and to gather information needed for identifying and classifying a unit for the catalog of compilation units the repository must support. The standard prologues are both compilable and SGML-processable. The SGML processable requirement allows the use of the full power of SGML to validate that text material meets a defined standard. Although there are five different prologues (the Design Header and Maintenance Header for non-nested compilation units, the Simple Design Header and Simple Subprogram Header for nested units, and the STARS Version Description for a released CSCI), one SGML DTD (Document Type *Definition*) describes them all. Given the DTD, a SGML parser that completely implements the SGML standard can validate that an input Ada compilation unit meets the requirements defined in that DTD. Once the input Ada compilation unit has been validated, the SGML-tagged information can easily be loaded into some database that is cataloging the compilation units stored on the repository.

3.3.1 Design Header

This prologue is to be used for the following non-nested units: standalone subprograms (compilation units) or main programs, package specifications (that have bodies), and generic instantiations.

```
--| -- This is a model of the prologue "Design Header"
--| -- which is to be used in accordance with the
--| -- Boeing STARS General S/W Development Plan,
--| -- CDRL 00670.
--|
--| -- Square brackets enclose optional items.
--| -- Braces enclose items which may be repeated
--| -- zero or more times.
```

```
--| [ --|Descriptive_Unit_Name>
--|       text ]

--|Description>
--| {text}
--||Description>

--|Requirements_Satisfied>
--| {text}
--||Requirements_Satisfied>

--|Revision_History>
--|Part_Number>   text
--|Author>        text
--| [--|Phone>    text]
--| [--|DDN>      text]
--|Version>       text
--|Change_Summary_Initial_Release>
--|Date>          text

--| {subsequent changes
--|Author>         text
--| {--|Phone>     text]
--| [--|DDN>       text]
--|Version>        text
--|Documents_associated_with_revision> text
--|Change_Summary> text
--|Date>           text
--|Release_Date>   text }
--||Revision_History>

--| [ --|Exceptions_Raised>
--|       {text}
--|   --||Exceptions_Raised> ]

--|Implementation_Dependencies_NONE>
--| -- use following if dependencies are located in the package body.
--|Warning_Contains_Dependencies>
--| -- use following if dependencies are located in the package spec.
--|Warning_Unit_Contains_Implementation_Dependencies>
--|Pragma_Usage>
--| {text}
--|Representation_Clause_Usage>
--| {text}
--|Hardware_Dependencies>
```

```
--| {text}
--|Compiler_Dependencies>
--| {text}
--||Warning_Unit_Contains_Implementation_Dependencies>

--|Implementation_Assumptions_NONE>
--| -- use following if implementation assumptions are located in
--| -- the package body.
--|Warning_Contains_Implementation_Assumptions>
--| -- use following if located in package specification.
--|Warning_Unit_Contains_Implementation_Assumptions>
--|Use_Assumptions>
--| {text}
--|Performance_Assumptions>
--| {text}
--||Warning_Unit_Contains_Implementation_Assumptions>

--|Distribution_and_Copyright>
--|   This prologue must be included in all copies of this software.
--|   This software is released to the Ada community.
--|   This software is released to the Public Domain (note:
--|   software released to the Public Domain is not subject to
--|   copyright protection).
--| Restrictions on use or distribution: NONE
--|
--|Disclaimer>
--|   This software and its documentation are provided "AS IS" and
--|   without any expressed or implied warranties whatsoever.
--|
--|   No warranties as to performance, merchantability, or fitness
--|   for a particular purpose exist.
--|
--|   Because of the diversity of conditions and hardware under
--|   which this software may be used, no warranty of fitness for
--|   a particular purpose is offered.  The user is advised to test
--|   the software thoroughly before relying on it.  The user
--|   must assume the entire risk and liability of using this
--|   software.
--|
--|   In no event shall any person or organization of people be
--|   held responsible for any direct, indirect, consequential
--|   or inconsequential damages or lost of profits.
--||Disclaimer>
```

3.3.2 Maintenance Header

This prologue is to be used for the following non-nested units: package specifications (that do not have bodies), subprogram bodies, package bodies, and task bodies.

```
--| -- This is a model of the prologue "Maintenance
--| -- Header" which is to be used in accordance
--| -- with the Boeing STARS General S/W
--| -- Development Plan, CDRL 00670.
--|
--| -- Square brackets enclose optional items.
--| -- Braces enclose items which may be repeated
--| -- zero or more times.
--|
--| -- Note: packages without bodies use this
--| --       prologue.


--| [ --|Descriptive_Unit_Name>
--|       text ]

--|Description>
--| {text}
--||Description>

--|Requirements_Satisfied>
--| {text}
--||Requirements_Satisfied>

--|Revision_History>
--|Part_Number>    text
--|Author>         text
--| [--|Phone>     text]
--| [--|DDN>       text]
--|Version>        text
--|Change_Summary_Initial_Release>
--|Date>           text

--| {subsequent changes
--|Author>          text
--| [--|Phone>      text]
--| [--|DDN>        text]
--|Version>         text
--|Documents_associated_with_revision> text
```

```
--|Change_Summary> text
--|Date>           text
--|Release_Date>   text }
--||Revision_History>

--| [ --|Exceptions_Raised>
--|       {text}
--|   --||Exceptions_Raised> ]

--|Implementation_Dependencies_NONE>
--| -- use following if dependencies are located in the package body.
--|Warning_Contains_Dependencies>
--| -- use following if dependencies are located in the package spec.
--|Warning_Unit_Contains_Implementation_Dependencies>
--|Pragma_Usage>
--| {text}
--|Representation_Clause_Usage>
--| {text}
--|Hardware_Dependencies>
--| {text}
--|Compiler_Dependencies>
--| {text}
--||Warning_Unit_Contains_Implementation_Dependencies>

--|Implementation_Assumptions_NONE>
--| -- use following if implementation assumptions are located in
--| -- the package body.
--|Warning_Contains_Implementation_Assumptions>
--| -- use following if located in package specification.
--|Warning_Unit_Contains_Implementation_Assumptions>
--|Use_Assumptions>
--| {text}
--|Performance_Assumptions>
--| {text}
--||Warning_Unit_Contains_Implementation_Assumptions>

--|Lessons_Learned>
--| {text}
--||Lessons_Learned>

--|Contact>
--| {text}
--||Contact>

--|Distribution_and_Copyright>
```

```
--|   This prologue must be included in all copies of this software.
--|   This software is released to the Ada community.
--|   This software is released to the Public Domain (note:
--|   software released to the Public Domain is not subject to
--|   copyright protection).
--| Restrictions on use or distribution: NONE
--|
--|Disclaimer>
--|   This software and its documentation are provided "AS IS" and
--|   without any expressed or implied warranties whatsoever.
--|
--|   No warranties as to performance, merchantability, or fitness
--|   for a particular purpose exist.
--|
--|   Because of the diversity of conditions and hardware under
--|   which this software may be used, no warranty of fitness for
--|   a particular purpose is offered.  The user is advised to test
--|   the software thoroughly before relying on it.  The user
--|   must assume the entire risk and liability of using this
--|   software.
--|
--|   In no event shall any person or organization of people be
--|   held responsible for any direct, indirect, consequential
--|   or inconsequential damages or lost of profits.
--||Disclaimer>
```

3.3.3 Simple Design Header

This prologue is to be used for the following nested units: package specifications, task specifications, generic (package) instantiations, package bodies, and task bodies.

```
--| -- This is a model of the prologue "Simple
--| -- Design Header" which is to be used in
--| -- accordance with the Boeing STARS General
--| -- S/W Development Plan, CDRL 00670.
--|
--| -- Square brackets enclose optional items.
--| -- Braces enclose items which may be repeated
--| -- zero or more times.


--| [ --|Description>
--|       {text}
--|   --||Description> ]
```

```
--| [ --|Exceptions_Raised>
--|       {text}
--|   --||Exceptions_Raised> ]
```

3.3.4 Simple Subprogram Header

This prologue is to be used for the following nested units: subprogram specifications, generic (subprogram) instantiations, and subprogram bodies.

```
--| -- This is a model of the prologue "Simple
--| -- Subprogram Header" which is to be used in
--| -- accordance with the Boeing STARS General
--| -- S/W Development Plan, CDRL 00670.
--|
--| -- Square brackets enclose optional items.
--| -- Braces enclose items which may be repeated
--| -- zero or more times.
```

```
--| [ --|Description>
--|       {text}
--|   --||Description> ]
.
--| [ --|Exceptions_Raised>
--|       {text}
--|   --||Exceptions_Raised> ]
```

3.3.5 STARS Version Description

This prologue is to be used for the Version Description Package which accompanies each CSCI release.

```
--| -- This is a model of the "STARS Version
--| -- Description" which is to be used in
--| -- accordance with the Boeing STARS General
--| -- S/W Development Plan, CDRL 00670.
--|
--| -- Square brackets enclose optional items.
--| -- Braces enclose items which may be repeated
--| -- zero or more times.
```

```
--|Revision_History>
```

```
--|Part_Number>    text
--|Author>         text
--|  [--|Phone>    text]
--|  [--|DDN>      text]
--|Version>        text
--|Change_Summary_Initial_Release>
--|Date>           text

--|  {subsequent changes
--|Author>         text
--|  [--|Phone>    text]
--|  [--|DDN>      text]
--|Version>        text
--|Documents_associated_with_revision> text
--|Change_Summary> text
--|Date>           text
--|Release_Date>   text }
--||Revision_History>

--|Class_I_Changes_Installed>
--|   List the new Class I changes to the CSCI/tool since the previous
--|   version/change.  Also identify for each entry in this list the
--|   related documentation (e.g., SPR) number and date.
--|
--|   This information does not apply to an initial release.
--||Class_I_Changes_Installed>

--|Class_II_Changes_Installed>
--|   List the new Class II changes to the CSCI/tool since the previous
--|   version/change.  Also identify for each entry in this list the
--|   related documentation (e.g., SPR) number and date.
--|
--|   This information does not apply to an initial release.
--||Class_II_Changes_Installed>

--|CSCI_Architecture>
--|   Identify and describe the static architecture of the CSCI/tool.  Booch
--|   or Buhr diagrams may be referenced.
--|   (STLDD 3.1)
--|
--|   Identify each compilation unit delivered with the CSCI/tool. Example:
--|   Nesting will identify each TLCSC, etc. begining with the TLCSC name
--|   followed by each of the names of the Ada compilation units that it
--|   withs (in compilation order). The CSCI must be identified by a part
--|   number.  In addition any standalone TLCSC must also be identified by
```

```
--|   a part number.  A standalone TLCSC is a driver that performs the
--|   function of a tool that is not nested in the main program of a CSCI.
--|   The part nuber is associated with the file containing the main program
--|   or top level package.
--||CSCI_Architecture>

--|Virtual_Interfaces>
--|   Identify the virtual interface(s) used by this CSCI/tool.
--|   (e.g., X Windows, GKS, CAIS, SQL).
--|   (SRS 3.3.1, 3.3.2)
--|   (SDDD 3.1, 3.1.x, 3.1.x.1, 3.1.x.2, 3.1.x.3)
--||Virtual_Interfaces>

--|Tool_Interfaces>
--|   Identify the tool interface(s) to this CSCI/tool.
--|   (SRS 3.3.3.1.X)
--|   (SDDD 3.1, 3.1.x, 3.1.x.1, 3.1.x.2, 3.1.x.3)
--||Tool_Interfaces>

--|Hardware_Interfaces>
--|   Identify the hardware interface(s) to this CSCI/tool.
--|   (SRS 3.3.3.2.Y)
--|   (SDDD 3.1, 3.1.x, 3.1.x.1, 3.1.x.2, 3.1.x.3) .
--||Hardware_Interfaces>

--|Functional_Control_and_Data_Flow>
--|   Describe the general high level flow of both control and data.
--|   A narative must be used to document the flow of data and control.
--|   The following diagraming techniques may be referenced: A control flow
--|   diagram may be used to represent control flow.  A data flow diagram
--|   may be used to describe the data flow.  If the tool is concurrent,
--|   then provide a finite state model diagram.  Buhr diagrams may be used
--|   to display control and data flow.
--|   (STLDD 3.4)
--||Functional_Control_and_Data_Flow>

--|System_Environment>
--|   Describe the environmental conditions under which this tool will
--|   operate.  (e.g., operating system and revision level, dependence on
--|   other languages, such as, FORTRAN, C, or Assembly routines or lib-
--|   raries.
--|   (SRS 3.5.1)
--|   This software was developed on the following hardware platform,
--|   operating system, Ada compilation system, (C compiler, and virtual
--|   interfaces, including: MIL-STD-1838 CAIS and X Windows Version 11R2.
```

```
--|
--|   Gould General Systems Division - NP1 Packaged System (SPU, 128 MB)
--|     Part No. 40518-021.
--|   Gould UTX/32 Operating System Rel. 3.1 (UNIX Berkeley 4.3 kernel
--|     with System V extensions).  Part No. 1470-0803.
--|   Gould APLEX Ada Compiler BSS (NP-1). Part No. 81040-2803.
--| Gould C Compiler with MCR Optimizer (NPL).  Part No. 1707-0803.
--|   Gould CAIS MIL-STD-1838 Prototype, Part No. 0006
--||System_Environment>

--|System_Parameters>
--|   Describe the parameters or constants that are used by the CSCI.
--|   Identify package specifications which document these constraints.
--|   (SRS 3.5.2)
--||System_Parameters>

--|System_Capacities>
--|   Describe and document the storage and execution requirements of the
--|   tool (e.g., memory and disk requirements).  Identify
--|   compilation units which are affected by these limitations.
--|   (SRS 3.5.3)
--||System_Capacities>

--|Installation_Instructions>
--|   Include an install file identifying the computer system and Ada
--|   compiler version that this batch install file was compiled on.  This
--|   will allow a user to cut this section out, delete the comment tokens
--|   and execute this batch command file to compile all of the sources
--|   required for the tool.
--|   (VDD 3.9)
--||Installation_Instructions>

--|Inventory_of_CSCI_Contents>
--|   Provide a cross reference of compilation units and their associated
--|   files by creating a list each of the ada compilation units in
--|   alphabetical order followed by the name of the file that contains
--|   the compilation unit.  (e.g., Some reusable software has more than
--|   one specification, etc. per file.  (e.g., package example · filename)
--||Inventory_of_CSCI_Contents>

--|Adaptation_Data>
--|   Identify and describe the data which is required during tool
--|   initialization.  These data should resemble the system environment
--|   and parameter information.  Also, identify the package specifications
--|   which implement these data.
```

```
--|   (STLDD 3.7)
--||Adaptation_Data>

--|Lessons_Learned>
--|   Enter lessons learned from the development of this tool.
--|
--|   1.  Verdix compiler does not recognize file names with greater than 24
--|       characters.
--||Lessons_Learned>

--|Classification_Facets>
--|Voice>
--|  Indicate the relation between the component and the operating
--|  environment.

--|Operating_System>
--|  Indicate the possible operating system in which the component can be
--|  invoked.

--|Lowest_Operating_System_Version>
--|  Indicate the lowest version level of operating system on which the
--|  component can be used.

--|Highest_Operating_System_Version>
--|  Indicate the highest version level of operating system on which the
--|  component can be used.

--|Compilation_System>
--|  List the compilation system(s) in which the component has correctly
--|  compiled.

--|Lowest_Compilation_System_Version>
--|  Indicate the lowest version level of operating system on which the
--|  component has been successfully compiled.

--|Highest_Compilation_System_Version>
--|  Indicate the highest version level of operating system on which the
--|  component has been successfully compiled.

--|Function>
--|  Describe the function of the component or of the component in which
--|  the component could be used.

--|External_Interface>
--|  Describe the form of external interface(s).
```

```
--|Application_Area>
--|  Describe the possible application area(s).

--|Entity>
--|  List the entities manipulated by the component.

--|Action>
--|  Describe the action performed on the entities.

--|Medium>
--|  Give the locale where the action is executed.

--|Domain>
--|  Describe the business, engineering, mathematical, scientific domain(s)
--|  in which the component can apply.
--||Classification_Facets>

--|Distribution_and_Copyright>
--|   This prologue must be included in all copies of this software.
--|   This software is released to the Ada community.
--|   This software is released to the Public Domain (Note:
--|     software released to the Public Domain is not subject
--|     to copyright protection).
--|   Restrictions on use or distribution: NONE
--|
--|Disclaimer>
--|   This software and its documentation are provided "AS IS" and
--|   without any expressed or implied warranties whatsoever.
--|
--|   No warranties as to performance, merchantability, or fitness
--|   for a particular purpose exist.
--|
--|   Because of the diversity of conditions and hardware under
--|   which this software may be used, no warranty of fitness for
--|   a particular purpose is offered.  The user is advised to test
--|   the software thoroughly before relying on it.  The user
--|   must assume the entire risk and liability of using this
--|   software.
--|
--|   In no event shall any person or organization of people be
--|   held responsible for any direct, indirect, consequential
--|   or inconsequential damages or lost profits.
--||Disclaimer>
```

3.3.6 SGML DTD for Prologues

The current version of the SGML DTD that would validate a prologue is:

```
<!DOCTYPE adacode [

<!-- The ADL (BNF) in the STARS Software Development Plan was
     used as the baseline.  Deviations are commented -->

<!ENTITY %assumptions_annotation
"(implementation_assumptions_none |
warning_contains_implementation_assumptions |
%assumptions_warning_annotation;)">

<!ENTITY %assumptions_warning_annotation
"(warning_unit_contains_implementation_assumptions)">

<!-- Design_Declarative_Part       ::== {design_basic_declarative_item}
                                       {design_later_declarative_item}
     Design_Later_Declarative_Item ::== (design_body |
                                        design_basic_subprogram_declaration |
                                        design_basic_package_declaration |
                                        design_basic_task_declaration |
                                        design_basic_generic_declaration |
                                        design_basic_generic_instantiation |
                                        #PCDATA)
     Design_Body                   ::== (basic_proper_body | #PCDATA) -->
<!ENTITY %body_stuff "((#PCDATA | basic_subprogram_body |
                        basic_package_body | basic_task_body |
                        design_basic_subprogram_declaration |
                        design_basic_package_declaration |
                        %design_basic_task_declaration; |
                        design_basic_generic_declaration |
                        design_basic_generic_instantiation)*)">

<!ENTITY %contact_annotation "(contact)">

<!ENTITY %copyright_disclaimer "(%copyright_disclaimer_annotation;)">

<!ENTITY %copyright_disclaimer_annotation
"(distribution_and_copyright, disclaimer)">
```

```
<!ENTITY %dependency_annotation
"(implementation_dependencies_none | warning_contains_dependencies |
%dependency_warning_annotation;)">

<!ENTITY %dependency_warning_annotation
"(warning_unit_contains_implementation_dependencies)">

<!ENTITY %description_annotation "(description)">

<!ENTITY %design_basic_task_declaration "(design_simple_task_declaration |
design_rendezvous_task_declaration)">

<!ENTITY %design_header "(%unit_name_annotation;?, %required_part;,
revision_history, %optional_part;?, %implementation_annotation;,
%copyright_disclaimer;)">

<!ENTITY %design_package_header "(%unit_name_annotation;?, %required_part;,
revision_history, %optional_part;?, %implementation_annotation;,
%lessons_learned_annotation;?, %contact_annotation;?, %copyright_disclaimer;)">

<!ENTITY %design_proper_body "(design_subprogram_body |
design_package_body | design_task_body)">

<!ENTITY %exceptions_annotation "(exceptions_raised)">

<!ENTITY %implementation_annotation
"(%dependency_annotation;, %assumptions_annotation;)">

<!ENTITY %initial_release_annotation
"(author, phone?, ddn?, version, change_summary_initial_release, date)">

<!ENTITY %lessons_learned_annotation "(lessons_learned)">

<!ENTITY %maintenance_header "(%unit_name_annotation;?, %required_part;,
revision_history, %optional_part;?, %implementation_annotation;,
%lessons_learned_annotation;, %contact_annotation;, %copyright_disclaimer;)">

<!ENTITY %optional_part "(%exceptions_annotation;)">

<!ENTITY %required_part "(%description_annotation;, %requirements_annotation;)">

<!ENTITY %requirements_annotation "(requirements_satisfied)">

<!ENTITY %simple_design_header "(%description_annotation;?, %optional_part;?)">
```

```
<!ENTITY %simple_subprogram_header "(%description_annotation;?,
%optional_part;?)">

<!-- Design_Basic_Declarative_Item ::== (design_basic_declaration | #PCDATA) -->
<!ENTITY %specification_stuff "((#PCDATA |
                                 design_basic_subprogram_declaration |
                                 design_basic_package_declaration |
                                 %design_basic_task_declaration; |
                                 design_basic_generic_declaration |
                                 design_basic_generic_instantiation)*)">

<!ENTITY %subsequent_change_annotation
"(author, phone?, ddn?, version, documents_associated_with_revision,
change_summary, date, release_date)">

<!ENTITY %unit_name_annotation "(descriptive_unit_name)">

<!--       ELEMENTS    MIN  CONTENT (EXCEPTIONS) -->

<!-- Design_Compilation_Unit ::== design_library_unit |
                                  design_secondary_unit |
                                  STARS_version_description
     Design_Secondary_Unit   ::== design_library_unit_body |
                                  design_subunit -->
<!ELEMENT adacode      - -  ((#PCDATA)*, (design_subprogram_specification |
                                          design_package_specification |
                                          design_generic_specification |
                                          design_generic_instantiation |
                                          design_subprogram_body |
                                          design_package_body |
                                          design_subunit) |
                                          STARS_version_description)>

<!ELEMENT design_subprogram_specification
                        - 0  (#PCDATA, %design_header;)>

<!-- Design_Package_Specification ::== (#PCDATA, design_package_header,
                                        {design_basic_declarative_item},
                                        [#PCDATA,
                                        {design_basic_declarative_item}],
                                        #PCDATA)
     Design_Package_Header        ::== (design_header | maintenance_header) -->
<!ELEMENT design_package_specification
                        - 0  (#PCDATA, %design_package_header;,
                              %specification_stuff;)>
```

```
<!ELEMENT design_generic_specification
                        - 0  (#PCDATA, (design_subprogram_specification |
                                        design_package_specification))>
<!ELEMENT design_generic_instantiation
                        - 0  (#PCDATA, %design_header;)>

<!-- Design_Subprogram_Body ::== (maintenance_subprogram_specification,
                                  [design_declarative_part], #PCDATA) -->
<!ELEMENT design_subprogram_body
                        - 0  (#PCDATA, %maintenance_header;,
                              %body_stuff;)>

<!-- Design_Package_Body ::== (#PCDATA, maintenance_header,
                                [design_declarative_part], #PCDATA) -->
<!ELEMENT design_package_body
                        - 0  (#PCDATA, %maintenance_header;,
                              %body_stuff;)>

<!ELEMENT design_subunit
                        - 0  (#PCDATA, %design_proper_body;)>

<!ELEMENT STARS_version_description
                        - 0  (#PCDATA, revision_history,
                              class_I_changes_installed,
                              class_II_changes_installed,
                              CSCI_architecture, virtual_interfaces,
                              tool_interfaces, hardware_interfaces,
                              functional_control_and_data_flow,
                              system_environment, system_parameters,
                              system_capacities, installation_instructions,
                              inventory_of_CSCI_contents,
                              adaptation_data, lessons_learned,
                              classification_facets, %copyright_disclaimer;,
                              #PCDATA)>

<!-- Design_Task_Body ::== (#PCDATA, maintenance_header,
                            [design_declarative_part], #PCDATA) -->
<!ELEMENT design_task_body
                        - 0  (#PCDATA, %maintenance_header;,
                              %body_stuff;)>

<!ELEMENT class_I_changes_installed
                        - -  (#PCDATA)*>

<!ELEMENT class_II_changes_installed
```

```
                        - -  (#PCDATA)*>

<!ELEMENT CSCI_architecture
                        - -  (#PCDATA)*>

<!ELEMENT virtual_interfaces
                        - -  (#PCDATA)*>

<!ELEMENT tool_interfaces
                        - -  (#PCDATA)*>

<!ELEMENT hardware_interfaces
                        - -  (#PCDATA)*>

<!ELEMENT functional_control_and_data_flow
                        - -  (#PCDATA)*>

<!ELEMENT system_environment
                        - -  (#PCDATA)*>

<!ELEMENT system_parameters
                        - -  (#PCDATA)*>

<!ELEMENT system_capacities
                        - -  (#PCDATA)*>

<!ELEMENT installation_instructions
                        - -  (#PCDATA)*>

<!ELEMENT inventory_of_CSCI_contents
                        - -  (#PCDATA)*>

<!ELEMENT adaptation_data
                        - -  (#PCDATA)*>

<!ELEMENT classification_facets
                        - -  (voice, operating_system,
                              lowest_operating_system_version,
                              highest_operating_system_version,
                              compilation_system,
                              lowest_compilation_system_version,
                              highest_compilation_system_version,
                              function, external_interface,
                              application_area, entity, action,
                              medium, domain)>
```

```
<!ELEMENT voice          - 0  (#PCDATA)>

<!ELEMENT operating_system
                         - 0  (#PCDATA)>

<!ELEMENT lowest_operating_system_version
                         - 0  (#PCDATA)>

<!ELEMENT highest_operating_system_version
                         - 0  (#PCDATA)>

<!ELEMENT compilation_system
                         - 0  (#PCDATA)>

<!ELEMENT lowest_compilation_system_version
                         - 0  (#PCDATA)>

<!ELEMENT highest_compilation_system_version
                         - 0  (#PCDATA)>

<!ELEMENT function       - 0  (#PCDATA)>

<!ELEMENT external_interface
                         - 0  (#PCDATA)>

<!ELEMENT application_area
                         - 0  (#PCDATA)>

<!ELEMENT entity         - 0  (#PCDATA)>

<!ELEMENT action         - 0  (#PCDATA)>

<!ELEMENT medium         - 0  (#PCDATA)>

<!ELEMENT domain         - 0  (#PCDATA)>

<!--       ELEMENTS    MIN  CONTENT (EXCEPTIONS) -->

<!ELEMENT design_basic_subprogram_declaration
                         - 0  (#PCDATA, %simple_subprogram_header;)>

<!-- Design_Basic_Package_Declaration ::== (#PCDATA,
                                            simple_design_header,
                                            {design_basic_declarative_item},
                                            [#PCDATA,
```

```
                                                [design_basic_declarative_item]],
                                                #PCDATA) -->
<!ELEMENT design_basic_package_declaration
                        - 0  (#PCDATA, %simple_design_header;)>

<!ELEMENT design_simple_task_declaration
                        - 0  (#PCDATA, %simple_design_header;)>

<!-- Design_Rendezvous_Task_Declaration ::== (#PCDATA, simple_design_header,
                                              #PCDATA) -->
<!ELEMENT design_rendezvous_task_declaration
                        - 0  (#PCDATA, %simple_design_header;)>


<!ELEMENT design_basic_generic_declaration
                        - 0  (#PCDATA, (design_basic_subprogram_declaration |
                                        design_basic_package_declaration))>

<!ELEMENT design_basic_generic_instantiation
                        - 0  (#PCDATA, (%simple_design_header; |
                                        %simple_subprogram_header;))>

<!-- Basic_Subprogram_Body ::== (design_basic_subprogram_specification,
                                 [design_declarative_part], #PCDATA) -->
<!ELEMENT basic_subprogram_body
                        - 0  (#PCDATA, %simple_subprogram_header;)>

<!-- Basic_Package_Body ::== (#PCDATA, simple_design_header,
                              [design_declarative_part], #PCDATA) -->
<!ELEMENT basic_package_body
                        - 0  (#PCDATA, %simple_design_header;)>

<!-- Basic_Task_Body ::== (#PCDATA, simple_design_header,
                           [design_declarative_part], #PCDATA) -->
<!ELEMENT basic_task_body
                        - 0  (#PCDATA, %simple_design_header;)>

<!--      ELEMENTS    MIN  CONTENT (EXCEPTIONS) -->
<!ELEMENT descriptive_unit_name
                        - 0  (#PCDATA)>

<!ELEMENT revision_history
                        - -  (part_number, %initial_release_annotation;,
                              %subsequent_change_annotation;*)>
```

```
<!ELEMENT contact      - -  (#PCDATA)*>

<!ELEMENT lessons_learned
                       - -  (#PCDATA)*>

<!ELEMENT implementation_dependencies_none
                       - 0  (#PCDATA)*>

<!ELEMENT warning_contains_dependencies
                       - 0  (#PCDATA)*>

<!ELEMENT implementation_assumptions_none
                       - 0  (#PCDATA)*>

<!ELEMENT warning_contains_implementation_assumptions
                       - 0  (#PCDATA)*>

<!ELEMENT description - -  (#PCDATA)*>

<!ELEMENT requirements_satisfied
                       - -  (#PCDATA)*>

<!ELEMENT exceptions_raised
                       - -  (#PCDATA)*>

<!ELEMENT warning_unit_contains_implementation_dependencies
                       - -  (pragma_usage, representation_clause_usage,
                             hardware_dependencies, compiler_dependencies)>

<!ELEMENT warning_unit_contains_implementation_assumptions
                       - -  (use_assumptions, performance_assumptions)>

<!ELEMENT part_number - 0  (#PCDATA)>

<!ELEMENT author       - 0  (#PCDATA)>

<!ELEMENT phone        - 0  (#PCDATA)>

<!ELEMENT ddn          - 0  (#PCDATA)>

<!ELEMENT version      - 0  (#PCDATA)>

<!ELEMENT change_summary_initial_release
                       - 0  (#PCDATA)*>
```

```
<!ELEMENT date          - 0  (#PCDATA)>

<!ELEMENT documents_associated_with_revision
                        - 0  (#PCDATA)>

<!ELEMENT change_summary
                        - 0  (#PCDATA)>

<!ELEMENT release_date
                        - 0  (#PCDATA)>

<!ELEMENT distribution_and_copyright
                        - 0  (#PCDATA)>

<!ELEMENT disclaimer    - -  CDATA>

<!ELEMENT pragma_usage
                        - 0  (#PCDATA)*>

<!ELEMENT representation_clause_usage
                        - 0  (#PCDATA)*>

<!ELEMENT hardware_dependencies
                        - 0  (#PCDATA)*>

<!ELEMENT compiler_dependencies
                        - 0  (#PCDATA)*>

<!ELEMENT use_assumptions
                        - 0  (#PCDATA)*>

<!ELEMENT performance_assumptions
                        - 0  (#PCDATA)*>
]>
```

3.3.7 Prologue Creation and Validation Tools

Interactive support for prologue creation ranges from incorporating an example and filling in the blanks with your preferred editor to context sensitive editors that only allow correct prologues to be created. CDRL 360, which describes SGML capabilities on the repository, discusses this range of editors more fully.

Non-interactive support for prologue creation includes tools that add a skeleton prologue to a compilation unit that doesn't have one and that analyze the unit's code for key constructs such as exceptions and pragmas, putting that information into the

proper area of the prologue. Boeing's Q Task 10 is specifying such a tool. It will aid the reuse of public domain source such as SIMTEL20, where prologues are either missing or incomplete.

Experiments with prologue validation were conducted using the current version of the STARS SGML parser. Firstly, the STARS parser does not completely implement the SGML standard, with the result that the prologue it can validate is not identical to the standard prologue. The examples below show the differences:

1. Markup tags used in Ada source code must start with "--" so that they will be treated as comments by the Ada compiler. The choice of delimiter characters is permitted through the use of an SGML declaration, but the STARS parser does not allow this declaration to be specified. Therefore, delimiters can only be altered by modifying the parser source code.

2. The optional markup minimization feature SHORTTAG is not implemented. SHORTTAG allows the tag close delimiter to be omitted from tags that are immediately followed by another tag. For example, the following excerpt from Design Header

```
--|Exceptions_Raised>
--| None.
--||Exceptions_Raised>

--|Implementation_Dependencies_NONE>

--|Implementation_Assumptions_NONE>

--|Distribution_and_Copyright>
--|   This prologue must be included in all copies of this software.
```

can be abbreviated to

```
--|Exceptions_Raised>
--| None.
--||Exceptions_Raised

--|Implementation_Dependencies_NONE

--|Implementation_Assumptions_NONE

--|Distribution_and_Copyright>
--|   This prologue must be included in all copies of this software.
```

SHORTTAG also allows empty tags. For example, Simple Subprogram

Header

```
--|Description>
--| An example of an empty tag.
--||Description>

--|Exceptions_Raised>
--| None.
--||Exceptions_Raised>
```

can be abbreviated to

```
--|Description>
--| An example of an empty tag.
--||>

--|Exceptions_Raised>
--| None.
--||>
```

3. The optional markup minimization feature SHORTREF is not implemented. With this feature, tags can be replaced by a single character or short strings. For example, the end tag

```
--||Warning_Unit_Contains_Implementation_Dependencies>
```

can be replaced by

```
--||
```

Secondly, the STARS SGML parser is too slow for actual repository use: a twenty-one line Ada program takes about four minutes to parse. Thirdly, the STARS SGML parser does not use the SGML DTD but a special form of Backus-Naur notation. This is related to its less than full support of SGML, with the consequence that maintaining a correct DTD is not sufficient to maintaining the prologue validation software.

We recommend that any SGML parser used by the repository to validate incoming compilation units implement the SGML standard completely.

4. Guidelines and Standards for Technical Documentation

The DoD Computer-Aided Acquisition and Logistic Support system (CALS) initiative aims to integrate the technical manuals, engineering specifications, etc. of the DoD and its suppliers into a homogeneous electronic system to provide for the "transition from current paper-intensive weapon support processes to the efficient use of digital information technology". To that end, it has created a standard for representing such information. That standard is the Automated Interchange of Technical Information - MIL-STD-1840A.

The "homogeneous electronic" philosophy of CALS is in tune with that of the STARS program and it makes a lot of sense for STARS to operate within the CALS framework. CALS is open in that it allows "foreign" things to fit within its framework and if they prove useful, they can become part of the CALS standard.

At this point, it looks like some parts of CALS, such as IGES and CCITT Group 4 for vector and raster graphics respectively, should be utilized directly by STARS. Other parts, such as document format, need to be tailored to better fit STARS needs. If such tailorings prove to have long-term, wide-spread value, they would be incorporated within the CALS standard.

4.1 Guidelines for Text

There seems to be a great deal of confusion about CALS and SGML. Many people use the two terms interchangeably - they are not. Part of the textual specification of CALS uses a subset of the SGML specification. SGML (ISO 8879) contains definitions and methods for marking "textual" material. The word textual has placed in quotes because SGML has been used for such things as musical scores, not obviously textual material. In Ada terminology, SGML would correspond to a "generic" and CALS would be a particular "instantiation" of that generic capability.

Preparation of the textual portion of a technical publication in an automated support environment can be viewed as a five step process:

a. creating a document type definition (DTD) for publication control;
b. authoring the publication and inserting SGML markup tags;
c. verifying that the syntax is correct according to the rules of SGML;
d. using the output specification to compose the document so that the produced (printed or displayed) copy corresponds to the proper format and style; and
e. generating a text presentation metafile in a page description language (PDL) to drive the display device, such as a printer or typesetter.

Current national and international non-government standards do not adequately address all five steps of the publication preparation process. ISO 8879 (SGML) addresses steps a, b, and c. No approved national or international standards exist to support steps d and e.

MIL-STD-1840A refers to MIL-M-28001 for handling textual information. MIL-M-28001 states that textual information shall be tagged per ISO 8879 and that there shall be Document Type Definitions (DTDs) and Output Specifications to define the tags and how to handle them. ISO 8879 is used to define the rules for the DTDs. MIL-M-28001 defines specific DTDs for MIL-M-38784B style documents. This style of document is used by the Air Force for some of its Technical Manuals e.g. a maintenance manual for a radio. Where MIL-M-38784B is not applicable, an alternative DTD must be prepared. Tagging and parsing are well defined. Various standards organizations around the world are working on developing Output Specification standards. Page Description Languages, such as Postscript, are in common use.

4.1.1 SGML Tag Sets and DTDs

There are two areas where the STARS program needs to do further work. The first is in the area of building a SGML tag set, with corresponding Output Specification, together with a set of Document Type Definitions (DTDs) for documents useful to people using STARS technology. Some of the tags defined for MIL-M-38784B could be used but several new ones are needed. The DTDs required are very different. Part of the STARS program direction is reuse. This implies not only reuse of code, but reuse of design, specifications, test, etc. SGML tags should be defined to ensure that information in a document can be properly entered and indexed in a database for later reuse.

As part of this task, a small DTD was developed for Technical Reports delivered during this period. It needs to be enhanced substantially to really implement the intent of SGML. It is reproduced here:

```
<!DOCTYPE doc [
<!ENTITY % text "( #PCDATA )" >
<!ELEMENT doc       - -  ( front? , body , rear? ) >
<!ELEMENT front      - -  ( idinfo , abstract? , preface? , contents , iluslist? ,
tablist? ) >
<!ELEMENT idinfo     - -  ( pubno , titleblk , addresee* , (mfr,contrno)* , pubdate
, ( sigblk* | sigpage ) ) >
<!ELEMENT pubno      - -  ( #PCDATA ) >
<!ELEMENT titleblk   - -  ( doctype? , prtitle , stitle? ) >
<!ELEMENT doctype    - -  ( #PCDATA ) >
<!ELEMENT prtitle    - -  ( #PCDATA ) >
<!ELEMENT stitle     - -  ( #PCDATA ) >
<!ELEMENT addresee   - -  ( #PCDATA ) >
<!ELEMENT mfr        - -  ( #PCDATA ) >
<!ELEMENT contrno    - -  ( #PCDATA ) >
<!ELEMENT pubdate    - -  ( #PCDATA ) >
<!ELEMENT sigblk     - o  ( ( purpose? , signer , position , organiz? , address? ,
```

```
date? ) | %text )  >
<!ELEMENT purpose   - o  ( #PCDATA )  >
<!ELEMENT signer    - o  ( #PCDATA )  >
<!ELEMENT position  - o  ( #PCDATA )  >
<!ELEMENT organiz   - o  ( #PCDATA )  >
<!ELEMENT address   - o  ( #PCDATA )  >
<!ELEMENT date      - o  ( #PCDATA )  >
<!ELEMENT sigpage   - o  EMPTY  >
<!ELEMENT abstract  - -  ( para+ , keywords )  >
<!ELEMENT keywords  - -  ( keyword* )  >
<!ELEMENT keyword   - -  ( #PCDATA )  >
<!ELEMENT preface   - -  ( para+ )  >
<!ELEMENT contents  - o  EMPTY  >
<!ELEMENT iluslist  - o  EMPTY  >
<!ELEMENT tablist   - o  EMPTY  >
<!ELEMENT body      - -  ( para0+ )  >
<!ELEMENT para0     - -  ( title , para* , subpara* )  >
<!ELEMENT subpara   - -  ( title , paratext? , para* , subpara* )  >
<!ELEMENT title     - -  ( %text; )  >
<!ELEMENT para      - -  ( paratext )  >
<!ELEMENT paratext  o o  ( %text; | list |  figure | table )*  >
<!ELEMENT list      - -  ( deflist )  >
<!ELEMENT deflist   - -  ( term | def )*  >
<!ELEMENT term      - -  ( #PCDATA )*  >
<!ELEMENT def       - -  ( %text; | paratext | table )*  >
<!ELEMENT figure    - -  ( graphic , title )  >
<!ELEMENT table     - -  ( title , graphic )  >
<!ELEMENT graphic   - o  EMPTY  >
<!ELEMENT rear      - -  ( appendix? , glossary? , index? , lep? , chgrec? )  >
<!ELEMENT appendix  - -  ( title | para0+ )*  >
<!ELEMENT glossary  - -  ( glosshd | deflist )*  >
<!ELEMENT glosshd   - -  ( #PCDATA )  >
<!ELEMENT index     - o  EMPTY  >
<!ELEMENT lep       - o  EMPTY  >
<!ELEMENT chgrec    - o  EMPTY  >
]>
```

4.1.2 SGML Tools

The second area is tools to support SGML usage. The tools will be separated into three types for discussion:

a. Parsers, Validators, etc. – tools that read and understand SGML tagged data
b. Automatic taggers – tools that automatically insert SGML tags into a file
c. Text Editors – tools that interactively aid in inserting SGML tags

4.1.2.1 SGML Parsers, Validators, etc.

A survey was conducted to determine what Parsers, Validators, etc. were available.

Sobemap of Belgium has a product called Markit. It was the first on the market and at the moment, appears to have the most capability. It can run as a standalone program for validation and/or to extract SGML information. It can also be used as a library of utilities in a user built program to do specialized processing. It does not have a good reputation but that appears to be as a result of its poor handling of errors. Typically, it does not find all the errors, the error messages are not helpful in defining what is wrong, and they refer to line numbers which are not helpful in determining the location of the error since lines in an SGML file don't correlate to lines in a document that produced it. The product is written in C and runs on MS-DOS and UNIX systems. Price is very dependent upon capability required and number of users. 10-25 thousand dollars might be typical for a modest sized group of users.

Software Exoterica of Canada is in the SGML business exclusively. It is a fairly new product but already has quite a reputation. They have capitalized on Sobemap's error handling problems by making error handling one of their strong points. It does not appear to be quite as complete as the Sobemap product. It can run as a standalone program for validation but the standalone cannot be used to extract SGML information. It can also be used as a library of utilities in a user built program to do specialized processing. The product is written in C and runs on VAX-VMS, UNIX and MacIntosh systems. Price is very dependent upon capability required and number of users. 8-20 thousand dollars might be typical for a modest sized group of users.

Datalogics of Chicago is entering the market. Their product is still in Beta test. It falls far short of fully supporting the ISO standard. It appears to be an entry level product to support CALS with their other documentation tools. It can run as a standalone program for validation but the standalone cannot be used to extract SGML information. It can also be used as a library of utilities in a user built program to do specialized processing. The product is written in C and runs on MS-DOS and VAX-VMS systems. Pricing is unknown.

IBM in Boulder has a new product but it is only available on IBM machines under the VM system and there do not appear to be any plans to expand its availability.

There is also an SGML "parser" developed under a STARS Foundations effort. This SGML processor validates that the input file conforms to the DTD for a specific document type. The document type is declared within the file. The SGML processor is a general-purpose parser, so it can not read an SGML DTD directly. The DTD is translated into a particular form of Backus-Naur notation, called BNF*. The processor uses the BNF* file to check the SGML tagged file for compliance. Its major difficulties are poor performance, poor error handling, not working directly with DTDs, and not fully supporting SGML.

4.1.2.2 Automatic SGML Taggers

In the area of automatic tagging, there appears to only be one contender – Avalanche Development Company of Boulder Colorado. Their IMSYS product tags an ASCII file that is formatted as a printed page. It only supports CALS documents. They are coming out with a new product called FASTTAG which will tag any style document in any form. They will supply the control data for it for CALS documents and this product will replace their IMSYS product. For non-CALS documents, you have to supply the control data.

4.1.2.3 SGML Text Editors

SGML editors are forthcoming from several companies. Context, Interleaf, Xyvision and most other large text editing companies with a DoD oriented clientele are adding SGML editing capabilities to their products. Both Sobemap and Software Exoterica have editors specifically tailored to SGML editing. In addition, there is an SGML editor developed under a STARS Foundation effort. It's tagging support works by expanding specific tags. It is not context sensitive, however. This means that any tag can be hand-created in any document section and that deletion of a opening or closing tag does not delete its partner or associated text.

The best approach would be to develop or obtain an SGML editing capability that was truly context sensitive so that syntactical tagging errors would be prevented. The transition to SGML tagging is more difficult than learning a new document formatting system since you can create tagging errors by misplacing or forgetting constructs just as you experience compilation errors in writing Ada code. In one sense, the problem is even larger than learning one programming language since many different DTDs will exist. Context sensitive editors exist for programming languages. The only difficulty here is to create a editor that used a given DTD to define its menus and rules for allowable constructs.

The Context DOC tool also demonstrates a long term direction SGML editing should go – that is, towards a WYSIWYG editor. This type of editor requires a workstation with a reasonable bit-mapped display. While it may be true that dumb terminals still represent the majority of video output devices available, the cost of workstations is declining in step with the rapid hardware advances.

4.2 Guidelines for Graphics

The CCITT Group 4 standard for raster graphics is a CALS standard that is fairly widely used as is IGES for vector graphics. Graphics delivered to the repository should be in one of these two formats. It should be noted that in both of these cases, the data describes the graphic and <u>not</u> the knowledge underlying the graphic. For example, while IGES data may contain information that a curved line is actually a circle, it does not contain any information that the circle represents a process in a data

flow diagram. Transfer if this information will require further investigation and development.

The *Boeing* Company

ACTIVE SHEET RECORD

SHEET NO.	REVLTR	ADDED SHEETS				SHEET NO.	REVLTR	ADDED SHEETS			
		SHEET NO.	REVLTR	SHEET NO.	REVLTR			SHEET NO.	REVLTR	SHEET NO.	REVLTR
1						30					
2						31					
3						32					
4						33					
5						34					
6						35					
7						36					
8						37					
9						38					
10						39					
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											
24											
25											
26											
27											
28											
29											

THE *BOEING* COMPANY

REVISIONS			
LTR	DESCRIPTION	DATE	APPROVAL